Clinical Natural Language Processing
Auto-Assigning ICD-9 Codes

Abe Coffman
School of Information
University of California, Berkeley
Berkeley, CA, USA
abecoffman@gmail.com

Nat Wharton
School of Information
University of California, Berkeley
Berkeley, CA, USA
wharton.n@gmail.com

Abstract

This document contains a summary of the authors' experiences with the Computational Medicine Center's 2007 Medical Natural Language Processing Challenge. It includes an overview of the basic NLP tasks they performed, their results, and suggestions for future work.

1 Introduction

The goal of this project was to develop an algorithm that could accurately auto-assign ICD-9-CM codes to clinical free text. The project itself is the final deliverable for the Applied Natural Language Processing[1] course taught by Barbara Rosario at the School of Information[2], UC Berkeley. The idea for the project, as well as the corpus, comes from the Computational Medicine Center's 2007 Medical Natural Language Processing Challenge.

1.1 Background

The World Health Organization (WHO) is responsible for the management of the International Classification of Diseases (ICD). The ICD is the international standard diagnostic classification for general epidemiological and clinical use.[3] The United States uses ICD-9-CM, a clinical modification of the 9th revision of ICD, to code and classify morbidity data from inpatient and outpatient records, and physician offices.[4] The National Center for Health Statistics (NCHS) and the Centers for Medicare and Medicaid (CMS) are the governmental organizations that oversee all modifications to ICD-9-CM in the United States.

While classification of morbidity data is useful for many reasons, the role ICD-9-CM codes play in healthcare reimbursement tends to overshadow other uses. Almost every patient encounter with a provider is tagged with one or more ICD-9-CM codes. These codes, in conjunction with Current Procedural Terminology (CPT) codes, are used to process the majority of health insurance claims in the United States. The Centers for Medicare and Medicaid negotiate the baseline reimbursement rates that other insurance providers benchmark against.

1.2 Purpose

Physician impressions and diagnoses are usually captured via dictation or as free text, either written by hand or entered into unstructured form fields. Coders, often professionally trained, must then interpret what the physician meant and assign ICD-9-CM codes prior to filing claims. Because the codes determine the amount the provider will be reimbursed for their service, it's crucial that they be as accurate as possible. The complexity of ICD-9-CM makes this task far from trivial.

As evidenced by our corpus, coders frequently disagree on what codes should be assigned. This is to be expected given the nature of interpretation tasks and the subtle differences between ICD-9-CM codes. While it's unrealistic to think that the right NLP algorithm could automate the process of assigning codes, algorithms can and already do help coders reach a decision faster by suggesting possible codes to choose from. The decision support approach has been embraced by the healthcare industry for a number of tasks, ICD-9-CM coding

[1] http://courses.ischool.berkeley.edu/i256/f09/
[2] http://www.ischool.berkeley.edu
[3] http://www.who.int/classifications/icd/en/
[4] http://www.cdc.gov/nchs/icd.htm

being just one of them. This project is an attempt at constructing a useful decision support algorithm for such a task.

2 Related Work

A number of the teams who participated in the Computational Medicine Center's challenge have since published their methods and the results they achieved. The strategies pursued by the authors of the two papers we've referenced were quite similar, and their algorithms both performed significantly better than ours. We will outline how their approach differed from ours here, and then briefly revisit some of the ideas that arise in our conclusion.

It became clear in reading the papers that possessing a strong understanding of the medical domain, and more specifically the logic behind the assignment of ICD-9-CM codes, is fundamental to this task. It also became clear that a purely statistical approach would underperform when put up against rule based systems. This point is well illustrated by considering the differences between the symptoms of diseases and the diseases themselves. To borrow an example from Crammer et al, the symptoms "cough" and "fever" are often associated with the disease "pneumonia." All three of these words have ICD-9-CM codes associated with them but the disease code will always take precedence if the patient has the disease. The challenge arises because in the majority of cases the patient will have these symptoms and not the disease. This leads to the over coding of symptoms.

Intelligent rule based systems have successfully overcome challenges like the one mentioned above. However these rules are tedious to manually code into your system. Farkas and Szarvas implement machine-learning techniques for discovering and implementing these rules, but this type of approach was beyond our scope. Instead we decided to see how well we could do with a purely statistical learning system that is described in more detail in sections three and four.

3 Data and Features

Our primary data source was a set of Radiology documents, marked up in XML, provided by the Computational Medicine Challenge. The documents were obtained from the Department of Radiology at the Cincinnati Children's Hospital Medical Center. Data was included in accordance with the following four principles:

1. All data was made completely anonymous in accordance with HIPPA standards (which included disambiguation and data scrubbing).
2. The dataset was constructed to best represent real-world conditions.
3. Data was also selected to represent all coding classes with more than one sample instance.
4. Representations of low-frequency classes were included.[5]

Each Radiology document contains two distinct sections. The first section is broken into two free text fields, the clinical history and the impression. The clinical history is a quick summary of a patient's condition as observed by the ordering physician prior to a radiological procedure. The impression is a Radiologists interpretation of the results of the procedure. The second section in each document contains a list of ICD-9-CM diagnostic codes relating to the patient visit.

3.1 ICD-9-CM Coding

ICD-9-CM codes range from 3-5 digits and are organized in a hierarchy. The general rule is that the more digits a code has, the more specific the clinical concept it represents. As mentioned previously, codes can represent both symptoms and diseases.

The rules for coding can be complex and sometimes arbitrary. As guidelines, certain diagnoses should always be coded and uncertain diagnoses should never be coded. In practice, coding can be highly variable and institution-specific. For this reason each document in the dataset contains codes from three separate institutions, the clinic and two external coding companies. In a democratic fashion, when at least two of the three sources chose the same code a "CMC Majority" code was added to reflect this agreement. This process is known as Majority Coding. It is possible for none of the three sources to agree (and thus provide no majority codes).

While each coder is allowed to assign multiple codes to a document, the first code they assign is

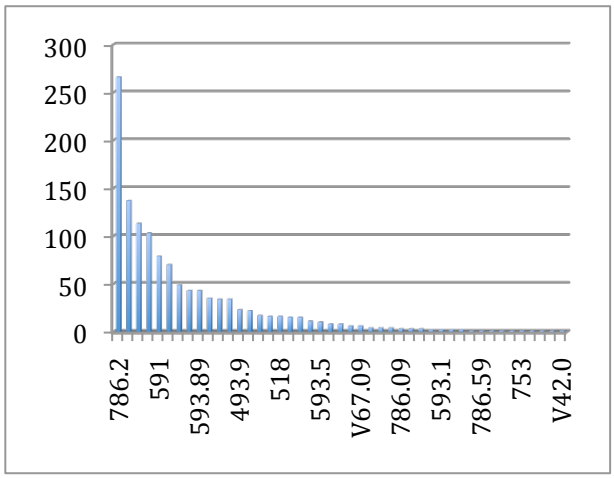
[5] Pestian JP, Itert L, Andersen CL, Duch W.

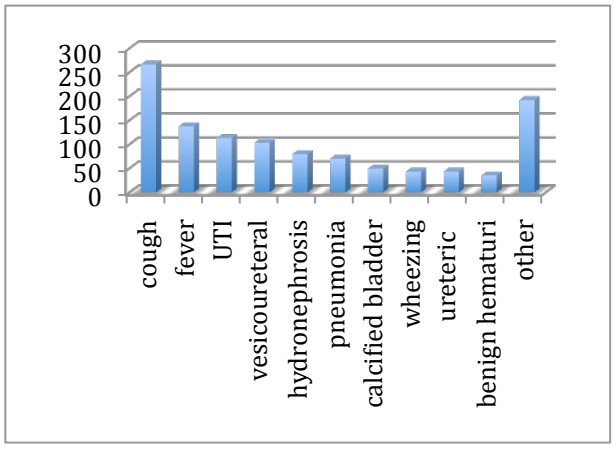
always considered the primary code. The primary code has the strongest weight of all the codes when considered for claim processing.[6]

The data from the challenge was split into the following two sets:

1. Training Set (978 documents)
2. Testing Set (976 documents)

The two sets contained a total of 45 unique ICD-9-CM codes. The distribution of these codes in the training set is shown below.





The documents also contained 94 distinct possible majority code combinations, though many of these combinations appeared no more than one or two times.

[6] Computational Medicine Challenge

3.2 UMLSKS Data

As we thought about how we might augment our dataset with external data it occurred to us that the most obvious improvement would simply be to leverage the official ICD-9-CM code definitions. In order to gain access to these definitions we had to register for a license from the United States National Library of Medicine (NLM). Once our license request was approved we were able to login to the Unified Medical Language Knowledge Source Server[7] (UMLSKS), or more specifically, the NLM's "UMLS and Source View" tool.

It turned out that the application had a bit of a learning curve due to the complexity of the data that it makes available. Eventually we were able to extract the definitions for all of the majority ICD-9-CM codes in our dataset. We then built a custom Python dictionary from the data using the IDC-9-CM codes as keys and the definition strings as values. This dictionary was used to build our corpus as described in the next section.

3.3 Corpus Building

In our first attempt at building a corpus we thought it would be helpful to subclass the XMLCorpusReader and XMLCorpusView, both included in the python-based Natural Language Toolkit (NLTK). As it turned out, the XML schemas upon which these libraries were based differed significantly from the XML documents in our primary data source. For instance, the British National corpus, the sole example[8] of these classes in use, is composed of numerous XML files, each of which contains their own set of documents. Most of the documents within these files are simply a single large text element. Our data, on the other hand, is composed of multiple documents included in a single XML file. Further, as mentioned previously, each radiology report document included two distinct sections, each with multiple child elements. In short, the given model didn't closely match our data source and attempting to make it work turned out to be more trouble than it was worth.

[7] http://www.nlm.nih.gov/research/umls/

[8] http://nltk.googlecode.com/svn/trunk/doc/api/nltk.corpus.reader.bnc-pysrc.html

Our second attempt proved to be successful because we parsed the XML ourselves and built our own custom corpus using python data structures, not something we had attempted in class previously. The function we developed successfully returns our custom corpus from base XML files for both training and test corpuses (and further works with subsets of the data that we used for development to avoid over-fitting of our data, etc…). We leveraged python's ElementTree module and used XPath to traverse the xml tree in building up the data structure.

At the core of our corpus there are the following two buckets:

1. Document Data (individual document data)
2. Corpus Data (aggregated document data)

All of the data we have either refers to a single document or to all of the documents in aggregate.

For each document in the document list we would extract, process and store the entire clinical history text and the entire impression text completely separately. We tokenized each text and then normalized it by lowercasing and stripping it. We then used NLTK's implementation of the Porter stemmer to stem each token, and again saved these lists of processed tokens separately.

For each set of tokens, we then removed all elements that were either punctuation, in NLTK's list of stop words, or were digits.

We further added part-of-speech (POS) tagging (using NLTK's POS-tagger) and saved the result of both the clinical history text and the impression text separately.

We further added bigrams for both clinical history and impression texts using NLTK's bigram processor.

For easy, standardized access in each document, we then also created a dictionary including all of the above information.

Code elements were then added to the corpus for each document. For each ICD-9-CM code, we stored the order of the code, the origin of the code (which designated the entity responsible for adding the code to the dataset).

For the dictionary of "overall corpus data", we stored data so it could easily be aggregated across all documents in one place. Overall corpus data that we created included all clinical history tokens, all impression tokens, the most common 75 history tokens, the most common 75 impression tokens, clinical history tagged with POS tagging, impressions tagged with POS tagging, all nouns in the clinical history, all nouns in the impressions, all clinical history bigrams, all impression bigrams, the 75 most common history bigrams and the top 75 most common impression bigrams.

Finally we added all stemmed, stripped, and lowercased (normalized) tokens from the UMLSKS definitions and added on the distinct set of its tokens to our corpus.

3.4 Features

We extracted a wide range of features from our corpus for our classification activities.

We extracted age as a feature via complex custom regular expressions from our corpus text sections found in radiation documents. We normalized age to an integer representing "number of days old", and then aggregated these integer-based ages into more coarsely-grained categories of ages: e.g. 'under 1 month old', 'up to 1 month old', 'under 3 months old', etc…).

We also indicated the presence (or lack of presence) of all UMLSKS definition tokens in both the history text and in the impression texts (set a Boolean value for each token).

We indicated categories with the number of words in each text description in case this was found to be significant.

We indicated the presence (or lack of presence) of the top 75 bigrams in both the clinical history text and in the impression text.

We indicated the presence (or lack of presence) of all nouns in our clinical history and impression texts.

We indicated the presence (or lack of presence) of top tokens in both our clinical history text and impression texts.

We also tried feeding back into our feature set the most informative words that we found from both our clinical history and impression texts, combined.

We ran many combinations of the above Features through our models. (You will see our results in section 4.4, below).

4 Models and Results

For our classification process we attempted to predict proper ICD-9-CM primary codes for all radiology documents in both the training and test corpuses using a variety of the features we extracted. In our case, primary codes were indicated as the code ordered first in association with a radiology document. Using this model we would achieve 100% accuracy if our predicted primary codes matched the actual primary codes found in the training and test datasets.

4.1 Naïve Bayes, Maximum Entropy, and Decision Tree Classification Models

We tested a wide matrix of features against Naïve Bayes, Maximum Entropy, and Decision Tree Classifiers. We experienced great success with the NLTK Naïve Bayes Classifier, some success with NLTK's Maximum Entropy Classifier (though its running literally took hours of processing time for single runs), and we experienced unrecoverable non-debuggable errors when attempting to run using NLTK's Decision tree classifier.

4.2 Runtime Constraints and Solutions

To attain the best results possible, we developed a matrix of feature possibilities for our classification tests. The matrix included more feature combinations than we had available compute time to possibly accomplish. In light of this constraint, we front-loaded many features into our corpus (as you may have noticed in the details of section 3.3, and then decided to serialize our pre-prepared corpus to disk using python's cPickle library. The corpus preparation phase that had previously taken minutes to compute now was reduced to mere seconds via the serialization technique, an encouraging accomplishment.

It was through repeated iterations that we achieved our final results matrix for the Naïve Bayes classifier.

4.3 Results

Note that running with the NLTK Maximum Entropy Classifier yielded generally lower results as seen in the charts, however, due to extremely long run times (even using pickling), it was not possible to attempt all the feature sets that we wished to use.

4.3.1 Naïve Bayes

Below are the results from some of the many feature set combinations we tried using Naïve Bayes:

Top 75 Words (H)	x	x	x	x		x			x	x		x
Top 75 Words (I)	x											
Most Informative (H)												
Most Informative (I)												
Word Count (I)				x								
Nouns (H)											x	
Nouns (I)											x	
Bigrams (H)					x	x	x					x
Bigrams (I)					x	x	x					x
Patient Age (H)			x	x	x	x				x		
ICD-9 Definitions (H)							x	x	x	x		x
ICD-9 Definitions (I)							x	x	x	x		x
Accuracy Score:	**76**	**76**	**76**	**76**	**63**	**73**	**71**	**72**	**77**	**77**	**51**	**75**

4.3.2 Maximum Entropy

Iteration	Log Likelihood	Accuracy
1	-3.63759	0.001
2	-1.38284	0.490
3	-0.97046	0.525
4	-0.78974	0.548
5	-0.68479	0.565
6	-0.61576	0.610
7	-0.56745	**0.619**
8	-0.53256	0.615
9	-0.50689	0.610

4.4 Benchmarks

The challenge assessed accuracy in terms of two f1 standards and a cost standard each defined mathematically in the competition documentation.[9] As previously noted, our accuracy was determined according to primary code accuracy. In spite of this, the competition accuracy results offer a useful benchmark to assess our work. The best f1 standard accuracy from the challenge was 89% and the average f1 standard accuracy in the challenge was 77%.

5 Conclusions

This project gave us the opportunity to apply the concepts we learned in our natural language processing course to an interesting domain. The challenge put forth by the Computational Medicine

[9] http://computationalmedicine.org/challenge/res.php

Center provided us a great corpus and a benchmark upon which we could measure our success. The National Library of Medicine gave us access to some valuable data with which to augment our corpus. The NLTK library provided us with the tools necessary to annotate the corpus, build our features, and test their effectiveness using some of the more common classification algorithms. To the extent that success is measured by how much you learn we're very happy with the results we obtained.

The applicability of the classification task we performed is indisputable. As discussed in section 1.2, there is a large and immediate commercial need for these types of algorithms. Not only could they decrease the costs of healthcare but they could also increase the accuracy of data that is used for all kinds of research, leading to improvements in quality and outcomes. These facts made our work more interesting and exciting.

The results we obtained using the material and the tools covered in class were pretty good. Using almost purely statistical methods and a multitude of different feature set combinations we were able to obtain an accuracy of 77%. While this was quite a bit lower than the ~90% accuracy that won the contest, it is significantly higher than what we would have achieved by simply assigning the most common ICD-9-CM code to everything. We are pretty confident that we could not have gotten much higher using our current methods.

There exists ample opportunity for future improvement to our algorithm. Addressing the symptom versus disease issue would probably be the best place to start. We could use some of our current machine learning methods to help us determine where we are over fitting and then implement custom rules to address these areas. This is essentially the approach taken by Crammer et al and Richárd Farkas and György Szarvas. However this is a time consuming process unless you can find a way to automatically discover and incorporate these rules.

Acknowledgments

We'd like to thank Barbara Rosario for teaching the natural language processing course and providing us with the background knowledge necessary to take on this project. We'd also like to thank Gopal Vaswani who provided assistance as the GSI for the course.

We should also mention that the Computational Medicine Center did a great job setting up this challenge and making the data available to students such as ourselves. We can only hope that more organizations such as this one will make corpus data available for future challenges in the medical field.

References

Computational Medicine Center. 2007. The Computational Medicine Center's 2007 Medical Natural Language Processing Challenge. http://computationalmedicine.org/challenge/index.php

Richárd Farkas and György Szarvas. 2007. *Automatic Construction of Rule-based ICD-9-CM Coding Systems*. BMC Bioinformatics, 2008. http://www.biomedcentral.com/content/pdf/1471-2105-9-S3-S10.pdf

Koby Crammer, Mark Dredze, Kuzman Ganchev, and Partha Pratim Talukdar. 2007. *Automatic Code Assignment to Medical Text*. http://www.cs.jhu.edu/~mdredze/publications/cmc_bionlp07.pdf

Pestian JP, Itert L, Andersen CL, Duch W. Preparing Clinical Text for Use in Biomedical Research. Journal of Database Management. 2005;17(2):1-12